DOCUMENT RESUME

ED 255 173 IR 011 542

AUTHOR Onaral, Banu; And Others
TITLE Interactive Computing and Graphics in Undergraduate Digital Signal Processing. Microcomputing Working Paper Series F 84-9.
INSTITUTION Drexel Univ., Philadelphia, PA. Microcomputing Program.
REPORT NO MWPS-F-84-9
PUB DATE Sep 84
NOTE 20p.; Paper presented at the Frontiers in Education Conference (Worcester, MA, October 17-19, 1983). For related documents, see IR 011 540-544.
AVAILABLE FROM Working Paper Series, Microcomputing Program, Office of the Vice President for Academic Affairs, Drexel University, 32nd and Chestnut Streets, Philadelphia, PA 19104.
PUB TYPE Reports - Descriptive (141) -- Speeches/Conference Papers (150)

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS *Computer Assisted Instruction; Computer Graphics; *Computer Science Education; *Courseware; *Design Requirements; *Engineering Education; Higher Education; Instructional Materials; Material Development
IDENTIFIERS *Digital Filter Design; Digital Signal Processing

ABSTRACT

This report describes the development of a Drexel University electrical and computer engineering course on digital filter design that used interactive computing and graphics, and was one of three courses in a senior-level sequence on digital signal processing (DSP). Interactive and digital analysis/design routines and the interconnection of these routines into a digital filter analysis/design package for computer assisted instruction (CAI) are discussed. Five examples illustrate the educational approach to integrating a software package into digital filter design coursework. The first two examples use a program based on the relationship between the pole-zero pattern of a given transfer function and its dynamic response. The design of digital filters from standard Butterworth and Chebyshev analog filters using the bilinear transform is explored in the third example, and the fourth and fifth examples illustrate the use of the computer aided methods proposed by Steiglitz and McClellan and Parks. A discussion of results based on a questionnaire survey of 68 students indicates that the CAI approach is effective in DSP. A majority of the students found the programs to be a worthwhile supplement to the course. Nineteen references are listed. (LMM)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED255173

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION
CENTER (ERIC)

☒ This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality.

• Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

Interactive Computing and Graphics in Undergraduate Digital Signal Processing

Banu Onaral, Davood Tashayyod, James M. Trosino
Department of Electrical & Computer Engineering
Drexel University

MWPS F 84-9
September, 1984

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

Banu Onaral

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

Requests for copies of papers in the Microcomputing Working Paper Series should be sent to: Working Paper Series, Microcomputing Program, Office of the Vice President for Academic Affairs, Drexel University, Philadelphia, PA 19104.

IR011542

BEST COPY AVAILABLE

BEST COPY AVAILABLE

Interactive Computing and Graphics in Undergraduate Digital Signal Processing*

Introduction

A sound understanding of discrete system concepts has become indispensable in the highly computerized environment of today. The versatility and affordability of microprocessors have expanded the range of digital signal and system applications in a multitude of high technology areas, particularly in electrical engineering. In order to meet the demands of industry and to stay competitive, engineering schools have introduced courses in Digital Signal Processing (DSP) at the graduate and advanced undergraduate levels. The present trend is toward the introduction of discrete system concepts earlier in the curriculum and in parallel with analog system theory[1].

The effectiveness of interactive computing and graphics in teaching theoretical concepts has long been recognized, demonstrated, and documented[2,3]. Topics of DSP are especially amenable to interactive presentation in a computer-aided instruction (CAI) setting. Successful curricula in various aspects of DSP have been developed on mainframes[4,5] or using microcomputers[6,7,8]. Best results are obtained when interactive computing and graphics are perceived as instructional tools designed to complement and enhance theoretical instruction.

The interactive dialogue between the user and the system entails fast and easy entry of specifications and/or data followed by immediate display, manipulation and comparison of results in a convenient form.

---

* Paper presented at the 1983 Frontiers in Education Conference, Worcester, Massachusetts, Oct. 17-19, 1983; Recipient of Benjamin Dasher Award for the best paper presented at the 1983 Frontiers in Education Conference.

Since students are freed from the rather routine and time consuming computational tasks, they can concentrate on the merits of a multitude of proposed solutions. This feature serves a major objective in conceptual learning, namely, the interpretation of the relationships between system parameters[3] which is a key factor in understanding analysis and design aspects of system theory.

Interactive computing and graphics is also the fundamental principle behind the novel "Computer-Aided Engineering (CAE) Workstation" concept. Workstations are perceived by industry as cost effective tools and a means to increase engineer's productivity[9]. Hence, an additional advantage to computer-aided approach is early exposure of students to this new technology with the ensuing competitive edge in the job market.

Topics of DSP are typically grouped into two major parts, digital filter design and spectral estimation with several overlapping areas. The senior-level DSP sequence of courses at Drexel University consists of three quarter-term courses covering the basics of discrete system theory, digital filter design and spectral estimation respectively. Each course lasts 10 weeks and is assigned 3 credit hrs/week, with 2 hrs. allocated to lectures and one to recitation period.

We present here our efforts on the Digital Filter Design part of the sequence. The project was undertaken in two stages:

- program development including both interaction and digital filter analysis/design routines.

- interconnection of these routines into a digital filter analysis/design package.

Our particular approach as well as factors we deem critical in establishing successful undergraduate courseware in DSP are also discussed. We have reported our work in the spectrum estimation part of the course in a more recent publication[10]. We are currently developing programs for the introductory systems courses on the Macintosh using the same approach. Manuscripts describing in detail these developments are

in preparation[11]. We have recently offered a workshop on developing interactive courseware for undergraduate discrete systems courses[12].

Resources

The team actively involved in the project consisted of the course instructor, one teaching assistant and an advanced senior student enrolled in the DSP course.

The computer used for the course (PRIME 850) became operational in Fall Term 1982 and supported a total of 70 televideo 925C alphanumeric terminals with 10 terminals dedicated for the exclusive use of the EE students. These terminals were upgraded with Retrographics Gen. II (RG1000/TU60) graphics cards from Digital Engineering to emulate Tektronix 4010 and 4027 series by Winter Term 1983. Keyboard entry and crosshair cursor were the only interaction modes available during the reported part of the project. For alphanumeric and graphics hard-copies, we have used a Printronix dot-matrix line printer and several Epson MX100 printer-plotters.

PRIME computer is operated under PRIMOS (version 19.1) operating system. The system software pool available for the project includes a host of compilers, utilities and scientific and application libraries. The compiler used in this project is structured FORTRAN 77. An Educational Software Package (ESP) prepared by Drexel Computer Center consists of utilities to assist in course coordination. Our software development benefited from two major libraries: IMSL which is a comprehensive set of mathematics and statistics routines and PRIME's Application Library which permits efficient coding for file-handling and I/O sections.

A comprehensive graphics software package is a major requirement of the project. However, such a package was not available during the initial part of the project. Instead, a restricted set of in-house plotting routines were used through the courtesy of the Physics Department. A commercial graphics software package "Template" was installed on PRIME in summer of 1983 and has been since integrated into our courseware.

Approach to Interaction

The goal of this project is to assist and facilitate the conceptual learning of DSP concepts. Interactive computing and graphics is a means best suited to this end. Therefore, the approach to interaction, both in terms of philosophy and methods, has critical implications in the success of the project. Unfortunately, guidelines for developing interactive instructional software are not apparent, due in major part to the large number and variability of alternatives going from one specific application to another[13]. The interaction scheme best suited to our situation gradually emerged as we experimented in the classroom environment. Note that the effort spent in arriving at the optimal scheme has so far exceeded the time devoted to scientific routine development. This has reflected into an approximate code ratio of 90% pertaining to user interaction in the final form of the package. We admit to having seriously underestimated this aspect at the onset of the project.

The package was integrated into coursework through three major facets: recitation demonstrations, homework and midterm problems. Component routines of the package were first introduced during hands-on recitation sessions under the guidance of the team members. They followed in-class discussion of the corresponding theory. A set of examples was prepared for each recitation. They served the dual purpose of stressing key points in related theory while exploring the features and capabilities of the new routines.

These routines were presented to the student in a menu-driven mode. At each branch point, the menu directs the attention of the user to a particular set of alternatives and/or decisions at hand allowing convenient selection of the most appropriate program route.

After the introduction of each routine, students were assigned homework problems designed to follow up on critical theoretical concepts. At this time, students had limited access to the course instructor and the assistants and they had to rely on the built-in HELP capability of the package.

During the first year we have presented a total of 7 recitation demonstrations. As we became more experienced in writing user-friendly software, in other words, as we improved upon our interaction scheme and HELP capability, routine introduction periods became less necessary judging from minimal help requested from the assistants in the last two recitations.

A few midterm examination problems also required the use of the package. Such problems were solved in two parts: an in-class part calling for an analytical solution to the problem and a subsequent take-home part in which students were allowed to test and/or improve the proposed solution by experimenting with the package.

Several examples illustrating our approach are discussed in the section on Software Development to follow.

Throughout this project, we have viewed student feedback as a most informative and valuable element in our courseware development. The validation process which consists of continually testing and revising programs offered to the students is a basic requirement of useful and healthy package development[13]. To this end, we have devised a practical student input mechanism whereby students could register their suggestions and complaints in a computer file or submit them as part of their homework solution. These recommendations were studied and implemented when appropriate.

Software Development

Educational software packages must conform to a set of basic criteria in order to become effective and long lasting teaching tools. To this aim, we have attempted to meet the following requirements:

(1) Comprehensiveness: The package must have the largest possible coverage of the same theoretical concepts treated in the course.

(2) Ease of use: Students use our programs to solve problems in order to experiment with theoretical concepts and their applications and not to improve their programming skills. An efficient, fast and

simple interaction scheme between the user and the programs is therefore a necessity. Sufficient intelligence to check and conveniently correct for simple data entry mistakes is also a desirable feature. Theoretical mistakes may also be detected but automatic correction would defeat the purpose of the package. Instead, the program must warn that a conceptual error has occured and require an acceptable entry.

(3) Modular structure: The package must be flexible enough to allow for future modification and expansion by addition of new routines.

(4) Transportability: Independence from hardware and operating systems must be sought.

The initial phase of the project consisted of developing various filter design routines which were eventually combined to form the package. Some of these routines were developed exclusively by the design team. Other programs were obtained from publications, previous senior design projects, and exchanges with other universities. All the routines received from other sources required modifications to conform to the critical user-interaction features of the package.

In the second phase we concentrated on the connection of all component subroutines into a Discrete System Design and Analysis (DSDA) Software Package. The filter analysis option is used to study the frequency and time response of an existing filter. The filter design option gives the user access to six different filter design algorithms contained in the package. The user selects a filter realization, enters filter specifications and is presented with a design solution. The dynamic response of the designed filter can be immediately studied both in the frequency and time domains and if necessary, original specifications may be changed or an alternate filter realization for the original specifications may be tried. The user may also elect to modify the parameters of the designed filter by interacting directly with the pole/zero pattern or the coefficient tables.

Examples

The examples below are selected to illustrate our educational approach to integrating a software package into digital filter design coursework and do not necessarily reflect the level of theoretical challenge presented to our students.

The first two examples use a program based on the relationship between the pole-zero pattern of a given transfer function and its dynamic response[14]. In our package, this program serves the dual role of frequency response analyzer and an empirical digital filter design tool. In the latter mode, it can be conveniently used to improve existing designs. The design of digital filters from standard Butterworth and Chebyshev analog filters using the bilinear transform is explored in the third example[15].

The fourth and fifth examples illustrate the use of popular "computer-aided" methods[16] proposed by Steiglitz[17] and McClellan and Parks[18].

Example 1. Understanding the relationship between the pole-zero pattern in the z-domain and the dynamic response is a first step in digital filter design. In the first recitation, the students explored this relationship by interacting with a program designed to conveniently manipulate the pole-zero pattern. The impulse, step and frequency responses of a first order low-pass filter were studied (Fig. 1-1). The effect of moving the pole toward the origin and reflecting it with respect to the origin was analyzed. (Fig. 1-2 and 1-3). Following this elementary example, students experimented with higher order and more complex patterns. By the end of the recitation, the students were expected to have sufficient understanding to propose an empirical design for a sixth order notch filter.

Example 2. This examination problem was intended to test the concepts introduced in example 1 above. The students were asked to carry out in-class the preliminary analytical work for a second-order digital filter designed to operate at a data rate of 500 samples/s. The desired specifications were 100% transfer at low frequencies with 3 dB cut-off at 25 Hz, 100% transfer at high frequencies with 3 dB cut-off at 225 Hz and no transfer in the region around 125 Hz. In the take-home part of the

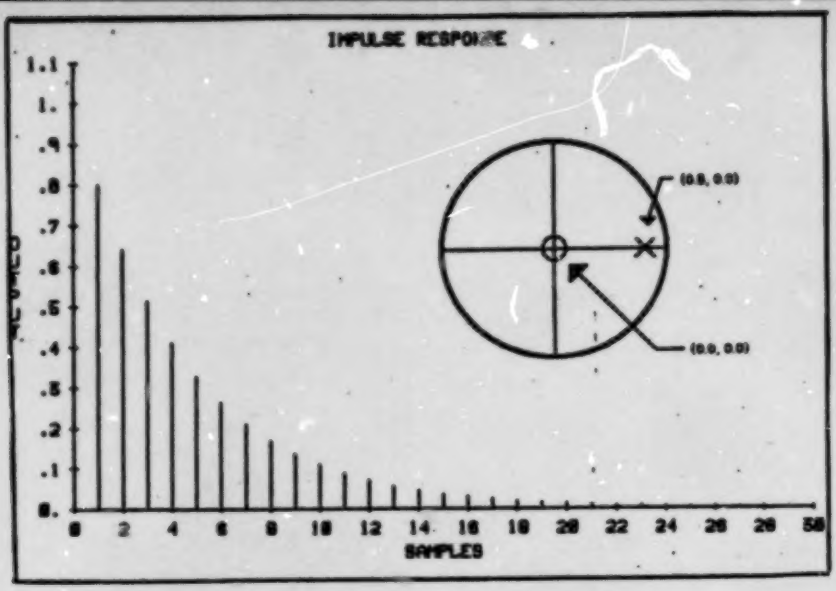

Fig. 1.1a

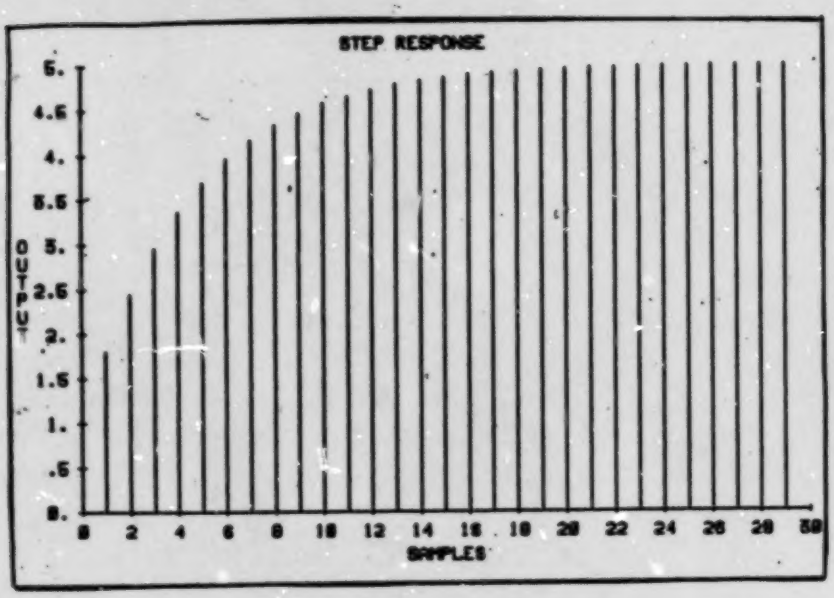

Fig. 1.1b

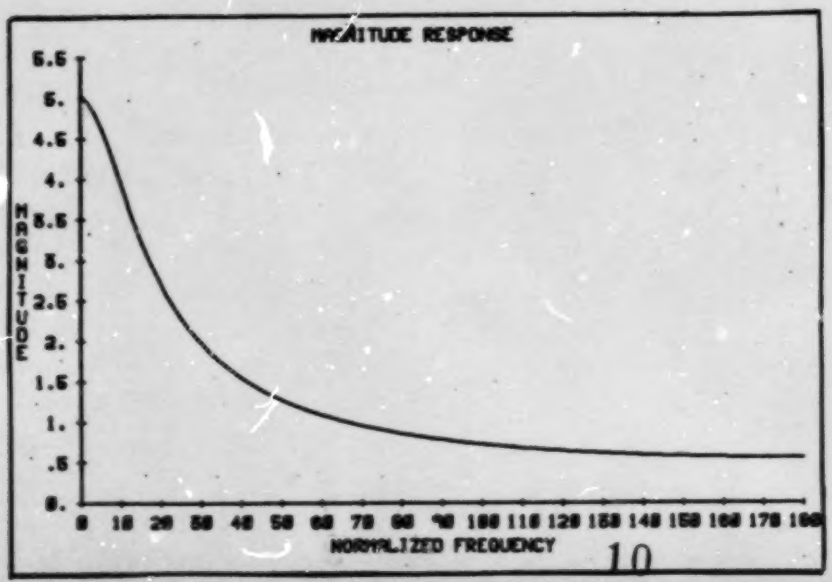

Fig. 1.1c

BEST COPY AVAILABLE

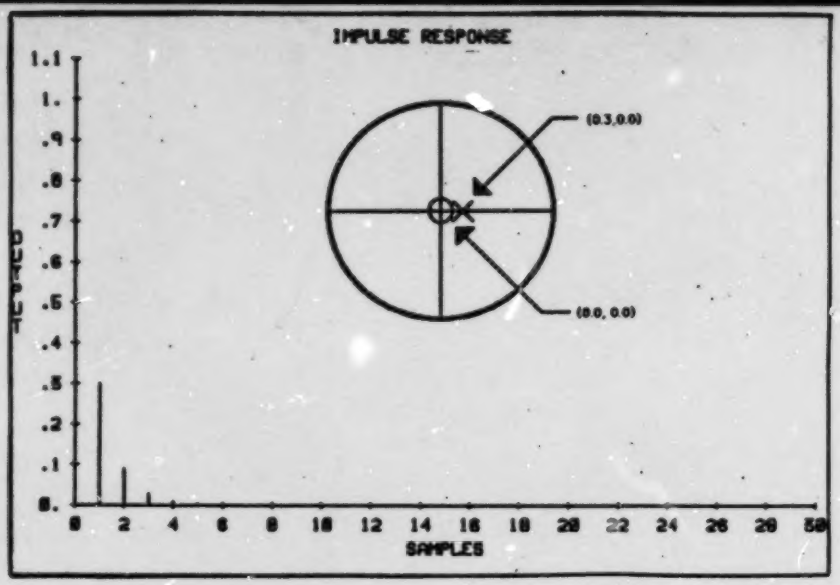

Fig. 1.2a

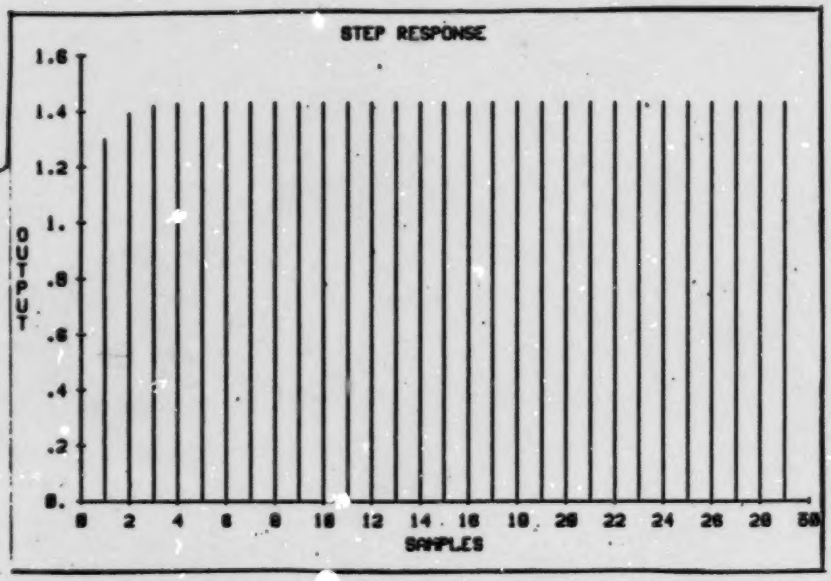

Fig. 1.2b

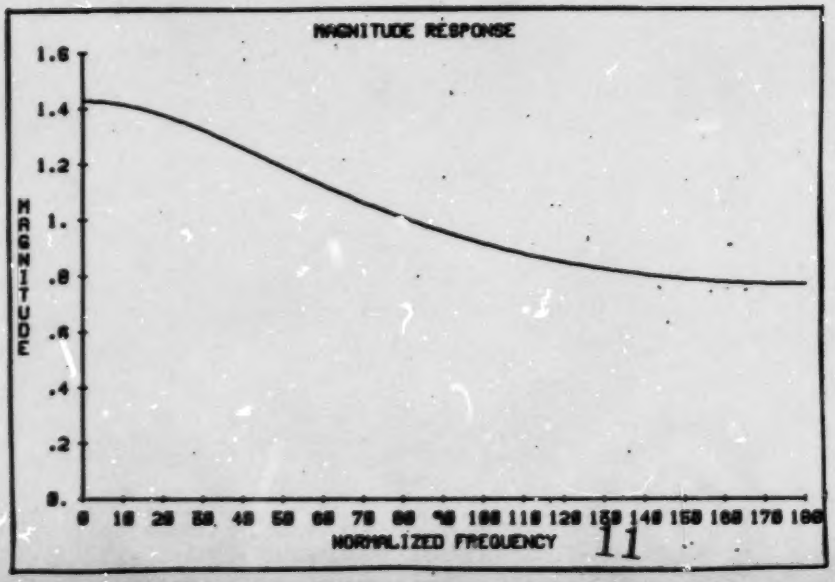

Fig. 1.2c

BEST COPY AVAILABLE

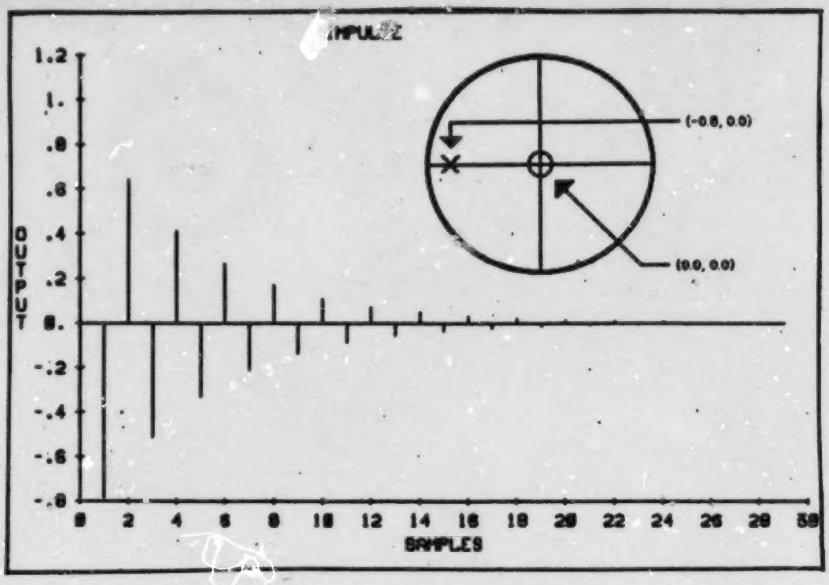

Fig. 1.3a

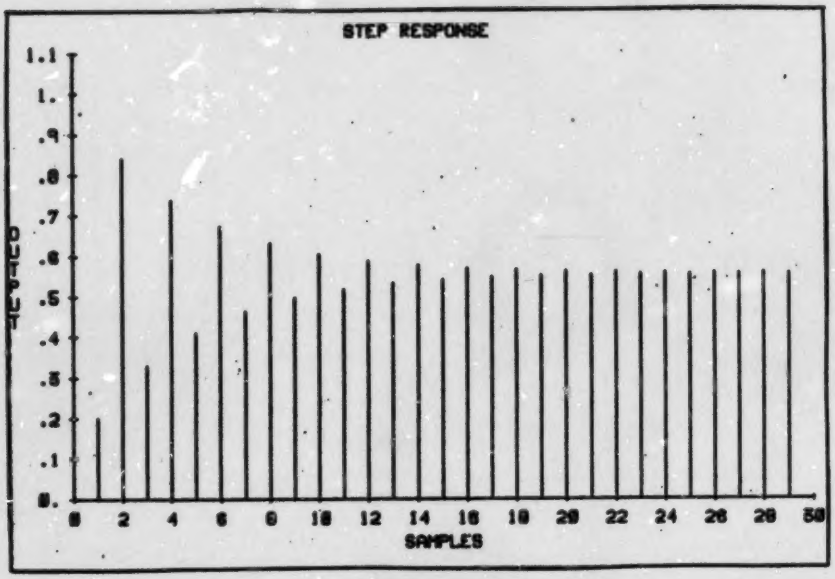

Fig. 1.3b

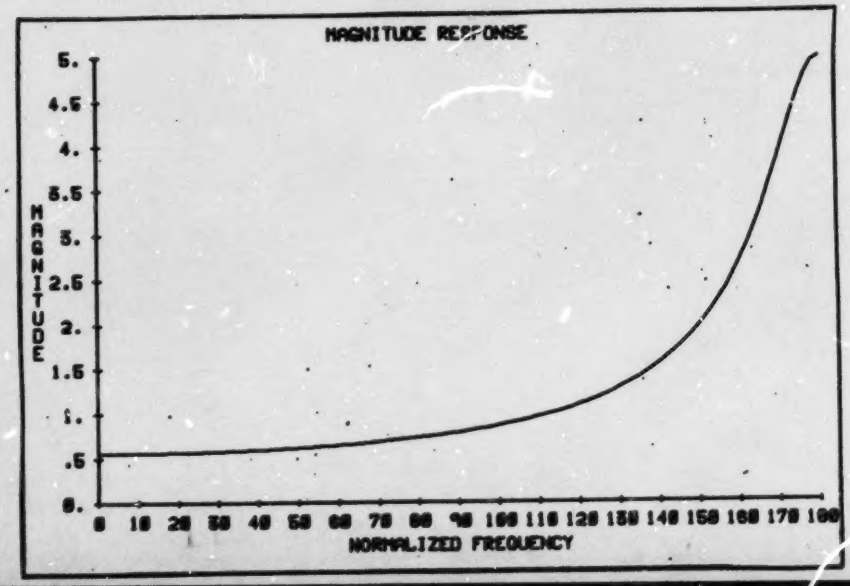

Fig. 1.3c

BEST COPY AVAILABLE

examination, students were asked to improve the proposed design by interacting with the package.

The unknown in this problem is the exact location of the poles on the real axis since the zeros must lie at the cross-section of the unit circle and the imaginary axis. The analytical solution for such a pattern is, however, a difficult task. An acceptable compromise is to move the zeros to the origin. The problem is then readily solved and approximate values of 0.69 are determined. Subsequent interaction with the program using these initial values suggests the correct position for the poles at 0.65 as shown in Fig. 2.

Example 3. It is beneficial for students to verify their analytical solutions by interacting with the package. As an examination problem, students were asked to do the paper-pencil design for the digital equivalent of an analog 1 dB between dc and 2 kHz and an attenuation of 60 dB or more beyond 7 kHz. The sampling frequency was 30 kHz. A sixth order Butterworth and a fifth order Chebyshev satisfied these specifications as verified by the program output for the Butterworth (Fig.3a) and the Chebyshev (Fig. 3b) filters.

Example 4. The effective use of a given design method implies a sound understanding of its strengths and weaknesses. During a recitation session students experimented with the recursive design suggested by Steiglitz[17] to realize a fourth order filter satisfying a set of point specifications (Fig. 4a). A remedy for an undesirable peak observed in the band from 60-90 Hz was required as a homework problem. The introduction of 4 more spec points in this band as opposed to manipulating the pole-zero pattern produced an acceptable solution (Fig. 4b).

Example 5. Students are expected to apply their knowledge about various design methods to practical problems. In a recitation, students were asked to design a filter to be used in decimating an oversampled (64 kHz) speech signal with a useful frequency content up to 3 kHz. They were expected to make an intelligent choice as to the type of the filter and the appropriate design method. A 128th order finite impulse response filter using the method by McClellan and Parks[18] with pass and stop band weights of 1 and 50 respectively is shown in Fig. 5.

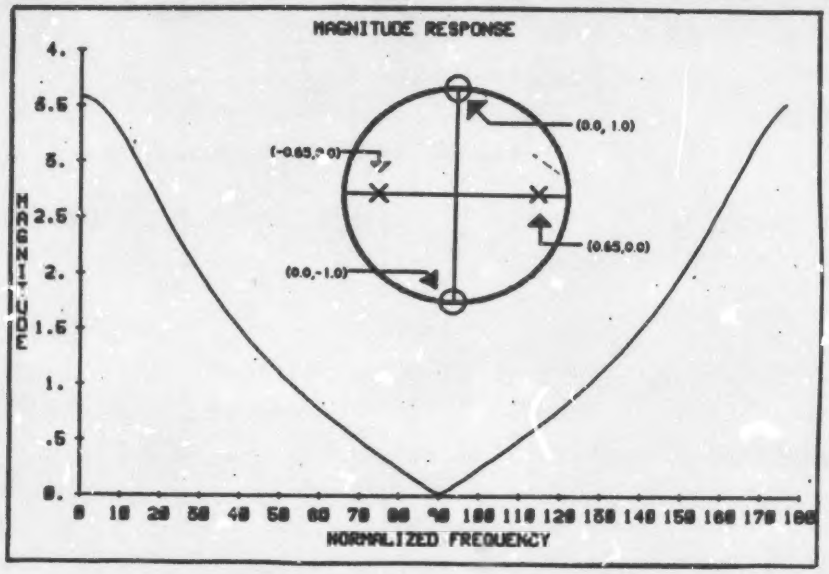

Fig. 2

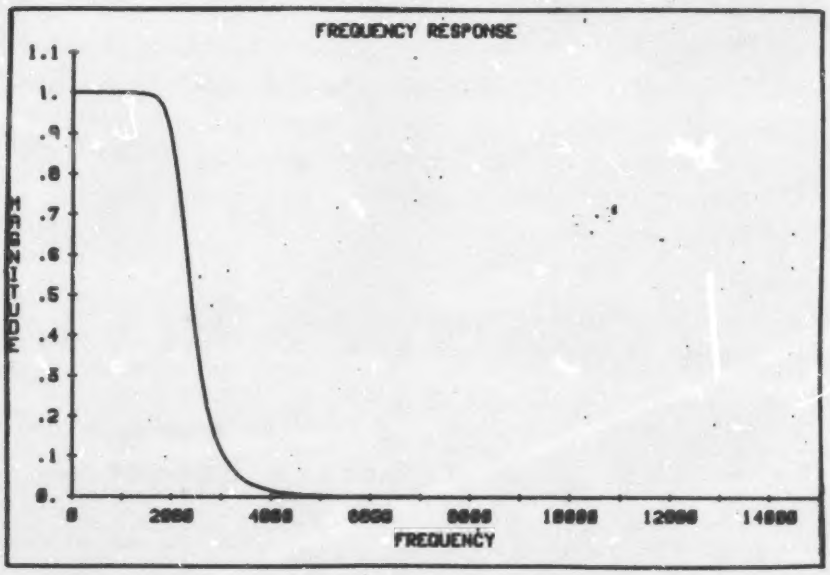

Fig. 3a

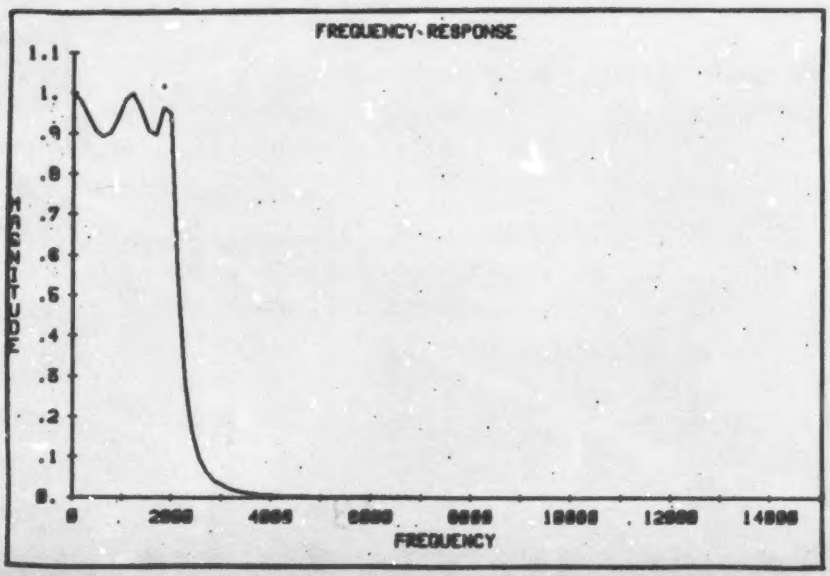

Fig. 3b

BEST COPY AVAILABLE

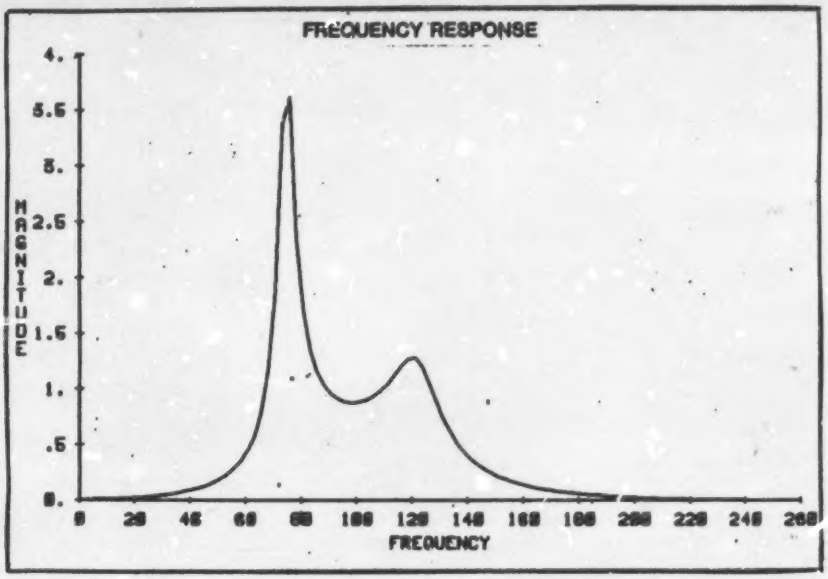

Fig. 4a

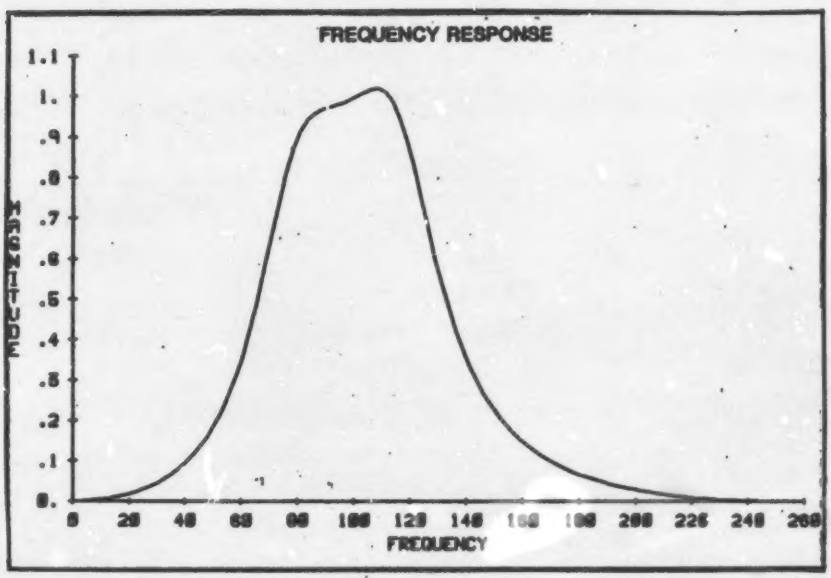

Fig. 4b

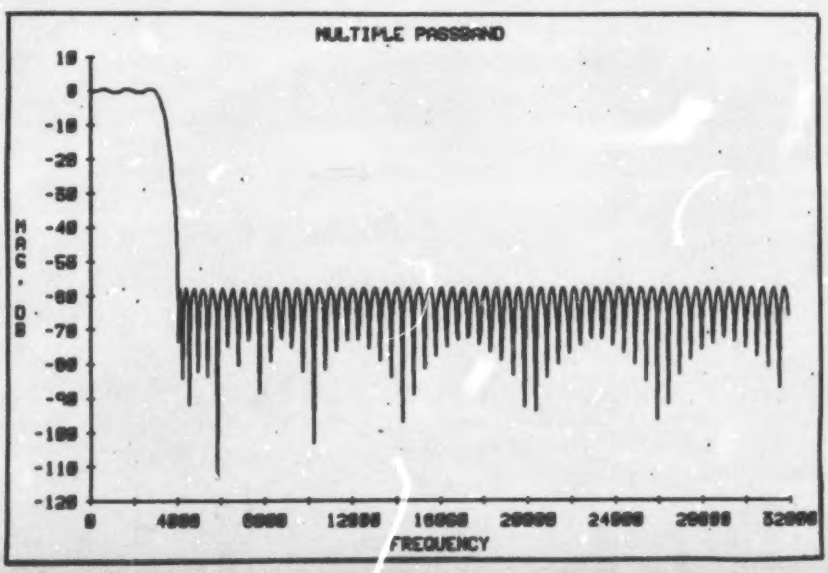

Fig. 5

BEST COPY AVAILABLE

Discussion

The development effort invested into this project over the academic year 1982-83 can be assessed by the size of the dedicated software as of the end of the Spring Term 1983. The total is estimated at 10,000 code lines corresponding to approximately 1,000 man-hours. This number compares well with the rule of thumb of 100 man-hours per instruction hour cited in CAI literature. It is clear that the project necessitates a major investment in manpower.

The effectiveness of CAI approach in DSP was evaluated in terms of the student response measured by a questionnaire administered in class to 68 students following the last recitation in the Spring Term 1983. These students had been enrolled in the entire sequence, thus, represented a group exposed to the progress of the entire project.

The favorable reception by the students of CAI in DSP is demonstrated in summary results compiled below.

1 * Over 88% of the students found the programs to be a worthwhile supplement to the course.

2 * Over 90% of the students said the DSP program should remain in the curriculum.

3 * Over 32% of the students said the DSP programs helped a great deal in understanding classroom theory; over 59% of the students remarked that the programs helped at least somewhat.

4 * Over 72% of the students said more programs should be introduced into the DSP curriculum.

The positive reception by the students is rewarding considering first-time difficulties peculiar to a pilot project which were further aggravated by the lack of quality interactive graphics hardware and software. These difficulties are reflected in the relatively poor response in item 3 above. This may be partly attributed to our inexperience in

BEST COPY AVAILABLE

efficient user-interaction schemes in the earlier stages of development, as well as occasional time lag between the presentation of the theory and the corresponding recitation demonstration owing to delays in completing programs.

In this presentation, we have shared our experience in initiating computer-aided instruction of a senior level digital filter design course. The advantage of integrating interactive computing and graphics into this curriculum is obvious. There is general agreement that this is the trend of the future. Yet, the pitfalls of hasty instructional innovation[19] as well as the risks, difficulties and costs involved in starting and successfully developing new programs must be recognized.

Our experience has shown that a commitment to the project on the part of both the institution and the faculty are critical to the success of the project.

Institutional commitment can be analyzed in terms of investment in system hardware and software as well as curriculum development. The acquisition and maintenance of the necessary facilities and the willingness of the administrative and technical staff to cooperate and collaborate with the faculty in matters of system selection and modes of operation is a strong indication of institutional commitment.

The investment in curriculum software development is in general severely underestimated.[3] The commitment in this regard can be measured by the generous allocation of the faculty and staff time to the project. The recognition of the development effort as equivalent of academic research activity may also serve to encourage the faculty.

The faculty in turn must demonstrate a willingness to deal with problems associated with the implementation of the project. A frustrating factor is the transient nature of the assisting staff in academia.[3] Teaching and research assistant assignments are usually much shorter than the period of development. Indeed, the variability in the background and the quality of students may further aggravate the situation. Software management in terms of development and maintenance of professional

BEST COPY AVAILABLE

quality documentation is necessary to ensure the continuity of the program but translates into added responsibility for the faculty. Specificity of the software to a given operating system, compiler and graphics hardware and software implies difficulties in transporting software. This impedes academic exchanges, acquisition of commercial packages and slows down the development process.

The decision to continue any pilot project hinges on accrued benefits weighed against the cost of development both in terms of facilities and man-hour. It must also take into account the critical factors as outlined above to ensure successful completion. In our case, most indicators pointed to the right direction.

BEST COPY AVAILABLE

References

1. Oppenheim, A. V., Willsky, A. S. Signals and Systems, Prentice Hall, 1983, Englewood Cliffs, N. J.

2. Musgrave, B. Training: CAD Courses for RPI Students. Datamation, 195-197, March 1980.

3. Wozny, M. J. The Rationale for Computer Graphics. Engineering Education. 73, 731-735, 1983.

4. Wozny, M. J., Frederick, D.K., Computer Graphics in Electrical Engineering at Rensselaer. Proc. ASEE Annual Conference. 201-213, 1981.

5. Prucnal, P. R. Experiments in Digital Signal Processing. IEEE Trans. E-25, 91-95, 1982.

6 Ziemer, R. E., Fannin, D. R., Stigall, P. D. Toward More Meaningful Laboratory Experiences in Digital Signal Processing and Digital Control. IEEE Trans. E-22, 119-123, 1980.

7. Bolton, A. G. Microprocessor Laboratory Procedure to Introduce Digital Filtering. IEEE Trans. E-24, 165-168, 1981.

8. Petersen, D. P. An Undergraduate Laboratory for Digital Control and Signal Processing. IEEE Trans. E-24, 168-173 1981.

9. Dinman, S. B. We've Seen the CAD - Now Let's See the CAE (Editorial). Computer Design. 22, 11, 1983.

10. Paarmann, L. D., Onaral, B. Interactive Computing and Graphics in Undergraduate Spectrum Estimation. 1984 Frontiers in Education Conference Proc., Philadelphia, PA, Oct. 305, 1984, 367-369.

11a. Paarmann, L.D., Onaral, B., Pushak, N. Computer-Aided Instruction in Undergraduate Digital Signal Processing, submitted to IEEE Trans. in Education.

11b. Pyeron, M., Onaral, B. Digital Filter Design and Analysis Package for Undergraduate Digital Signal Processing Courses; in preparation.

11c. Tanyel, M., Nahum, A., Onaral, B. Signal Operations Courseware for Introductory Systems Engineering Courses; in preparation.

BEST COPY AVAILABLE

12. Onaral, B., Paarmann, L. D. Developing Interactive Computing and Graphics for Undergraduate Discrete Systems Courses. 1984 Frontiers in Education Conference Proc., Philadelphia, PA, Oct. 3-5, 1984, 367-369.

13. Burke, R. L. CAI Sourcebook, Englewood Clifs, N. J., Prentice Hall. 1982.

14. Cadzow, J. A. Discrete-Time Systems, Englewood Cliffs, N. J., Prentice Hall, 1973, ch.10.

15. The original source code of this program was obtained with the courtesy of K. Young of the University of Wisconsin-Madison, Electrical and Computer Engineering Department.

16. Oppenheim, A.V., Schafer, R.W. Digital Signal Processing, Englewood Cliffs, N. J., Prentice Hall, 1975.

17. Steiglitz, K. Computer-Aided Design of Recursive Digital Filters, IEEE Trans. AU-18, 123-129, 1970.

18. McClellan, J. H., Parks, T. W., Rabiner, L. R. A Computer Program for Designing Optimum FIR Linear Phase Digital Filters, IEEE Trans. AU-21, 506-525, 1973.

19. Rainey, G. L. How to Survive Instructional Innovation, Engineering Education, 72, 154-157, 1981.